BIPS

C O M P U T A T I O N A L   R E S E A R C H   D I V I S I O N

Architecture Characterization using APEX-Map

Erich Strohmaier
Future Technology Group
EStrohmaier@lbl.gov

SC2005, Nov. 15, 2005

Office of Science
U.S. DEPARTMENT OF ENERGY

Outline

- APEX-Map principals and previous sequential results
- Parallel APEX-Map and results
- APEX-Map possible extensions, current and future work

Why? What? How?

- **Application performance is what we care about most.**
  - Using real applications for performance work can be very tedious.
- **Synthetic benchmarks are much easier.**
  - They are great for architectural studies.
  - But we generally don't understand how the performance of synthetic benchmarks relates to applications!
- **Is there a methodology to create synthetic benchmarks, which capture the main performance effects of real applications?**
  - Are there micro benchmarks which characterize applications?

Application Performance Characterization (Apex)

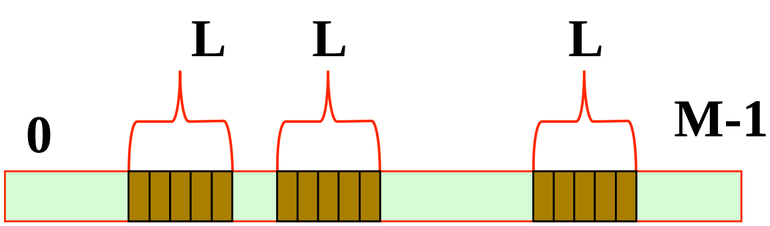
- Initial focus was the performance influence of global data-access.
- We view data access as composed of one or multiple data access streams.
- We characterize data access streams independent of each other.
- We try to use as few streams as possible (one).

Performance Aspects of Data Access Streams

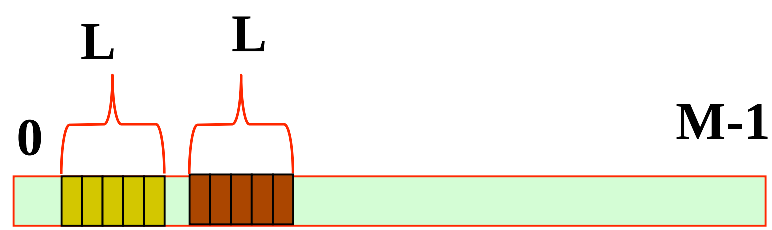
- **Regularity – 2 extremes:**
  - Random walk in memory
  - Regular advance in memory (strided access)
- **Data set size (*M*)**
- **Length of contiguous data access (*L*)**
  - Vector Length, Spatial Locality
- **Temporal Locality (α)**
  - Characterized by the exponent α of a approximation of the temporal distribution function by a power law.

Memory Access Probe
Apex-MAP

- For *Random* access (start of vectors) we choose:
  - Use the Power distribution for the non-uniform random address generator.
    - Self-similar and thus scale invariant.
    - Exponent α in [0,1]
      - $\alpha=1$: Uniform random access.
      - $\alpha=0$: Access to a single vector only.



$\alpha \approx 1$



$\alpha \approx 0$

Apex-Map Inner Loop
Random Access Stream

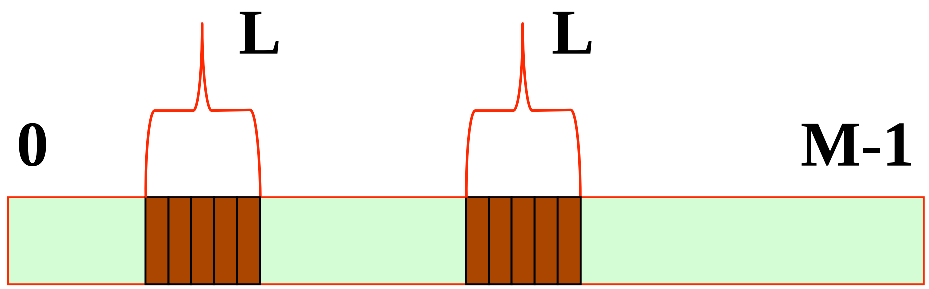
BIPS

```
for ( i = 0; i < N; i++) {
    initIndexArray(I);
    CLOCK(time1);
    for (j = 0; j < I/4; j++) {
        pos  = ind[j*4];
        …
        for (k = 0; k < L; k++) {
            res0 += data[pos  + k];
            …
        }
    }
    CLOCK(time2);
}
```

Initialize addresses

Unrolled four times

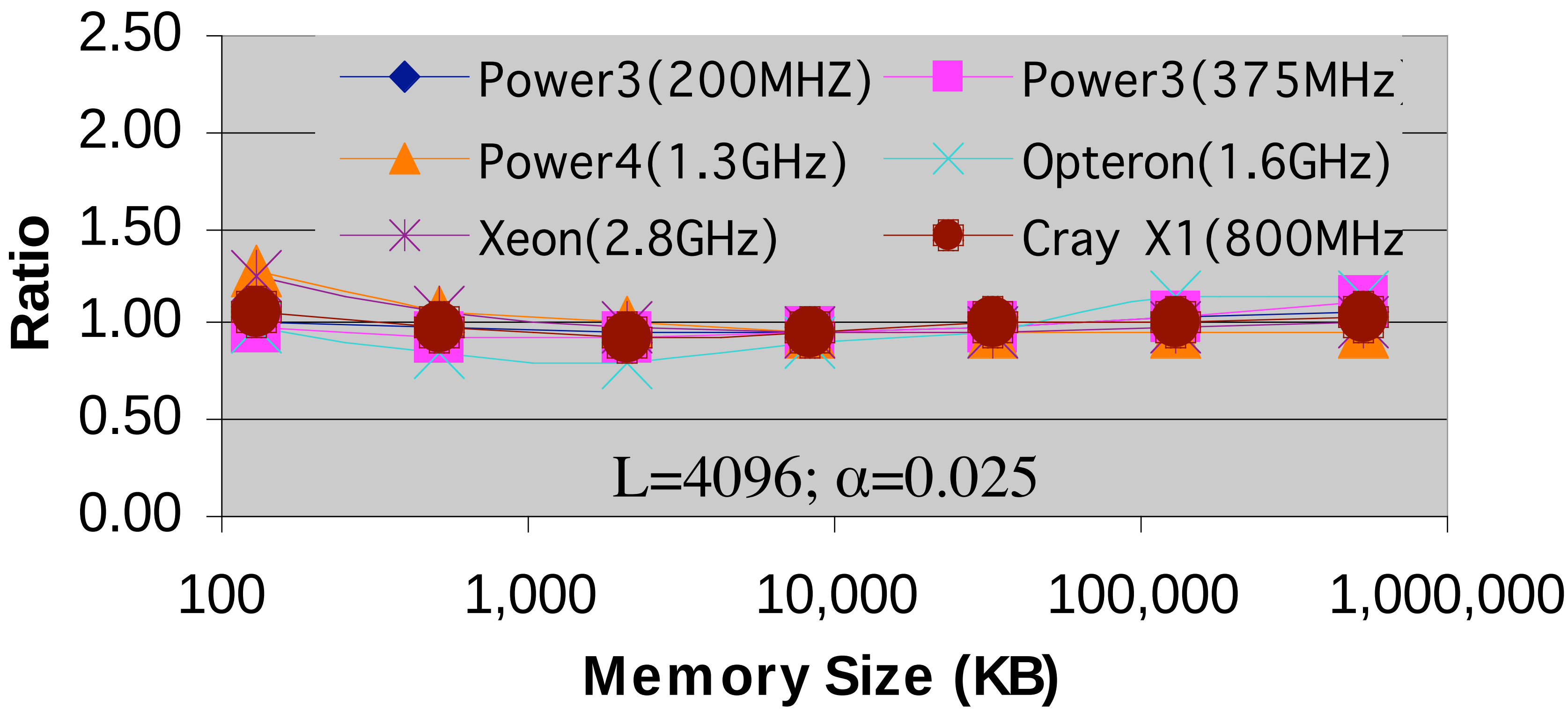
Vector access



Office of Science
U.S. DEPARTMENT OF ENERGY

BERKELEY LAB

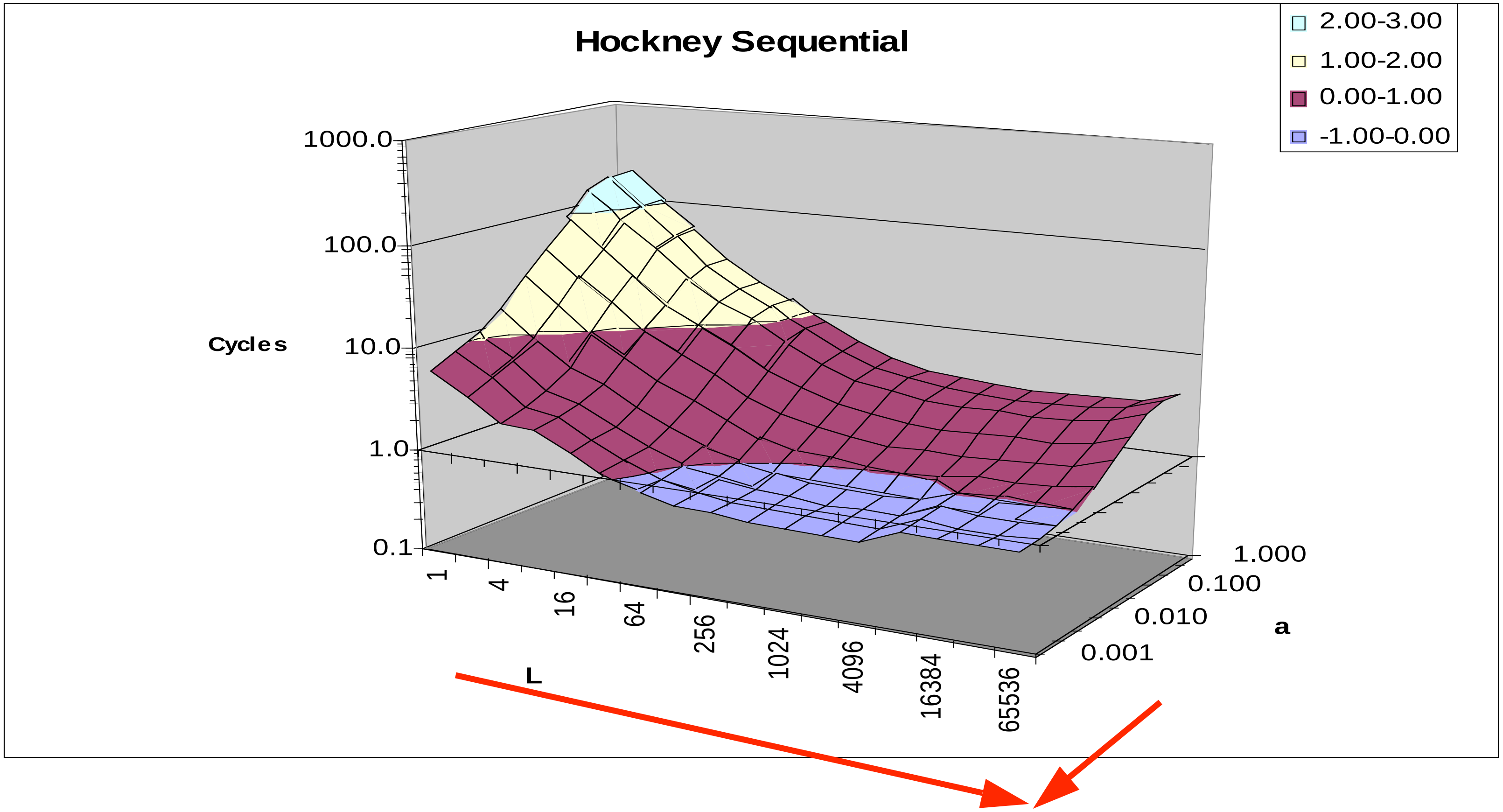
APEX-Map and Nbody



- **Using one random access stream**

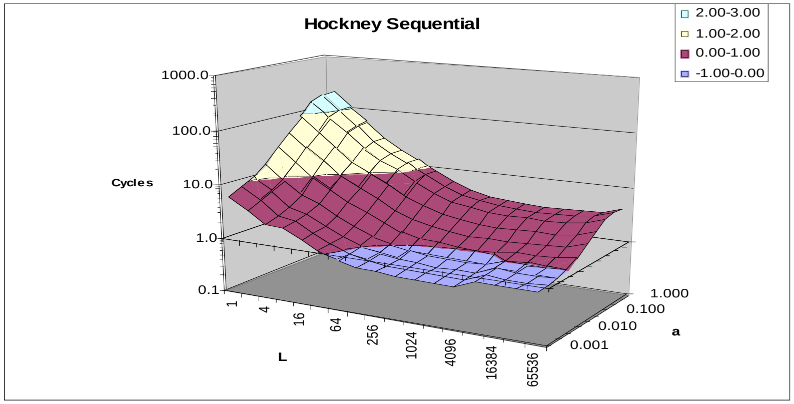
Published at MASCOTS 2004

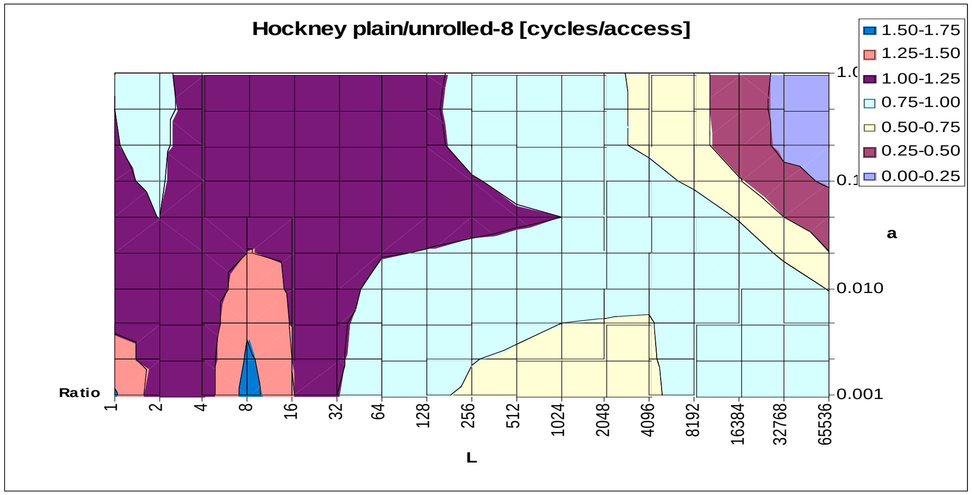
Apex-Map Sequential

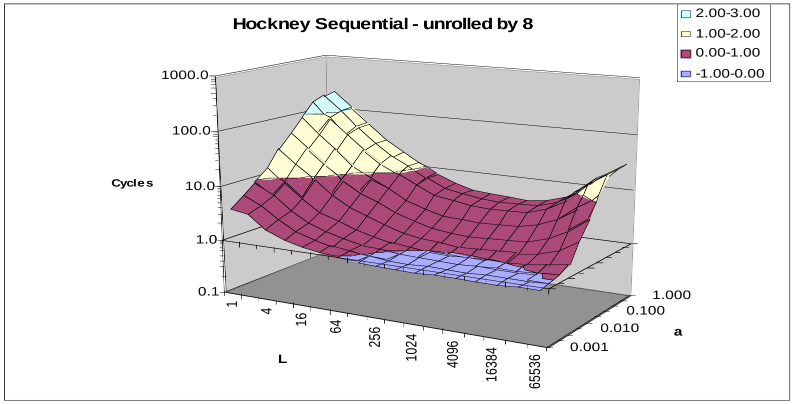


Power3 375 MHz

Loop Unrolling

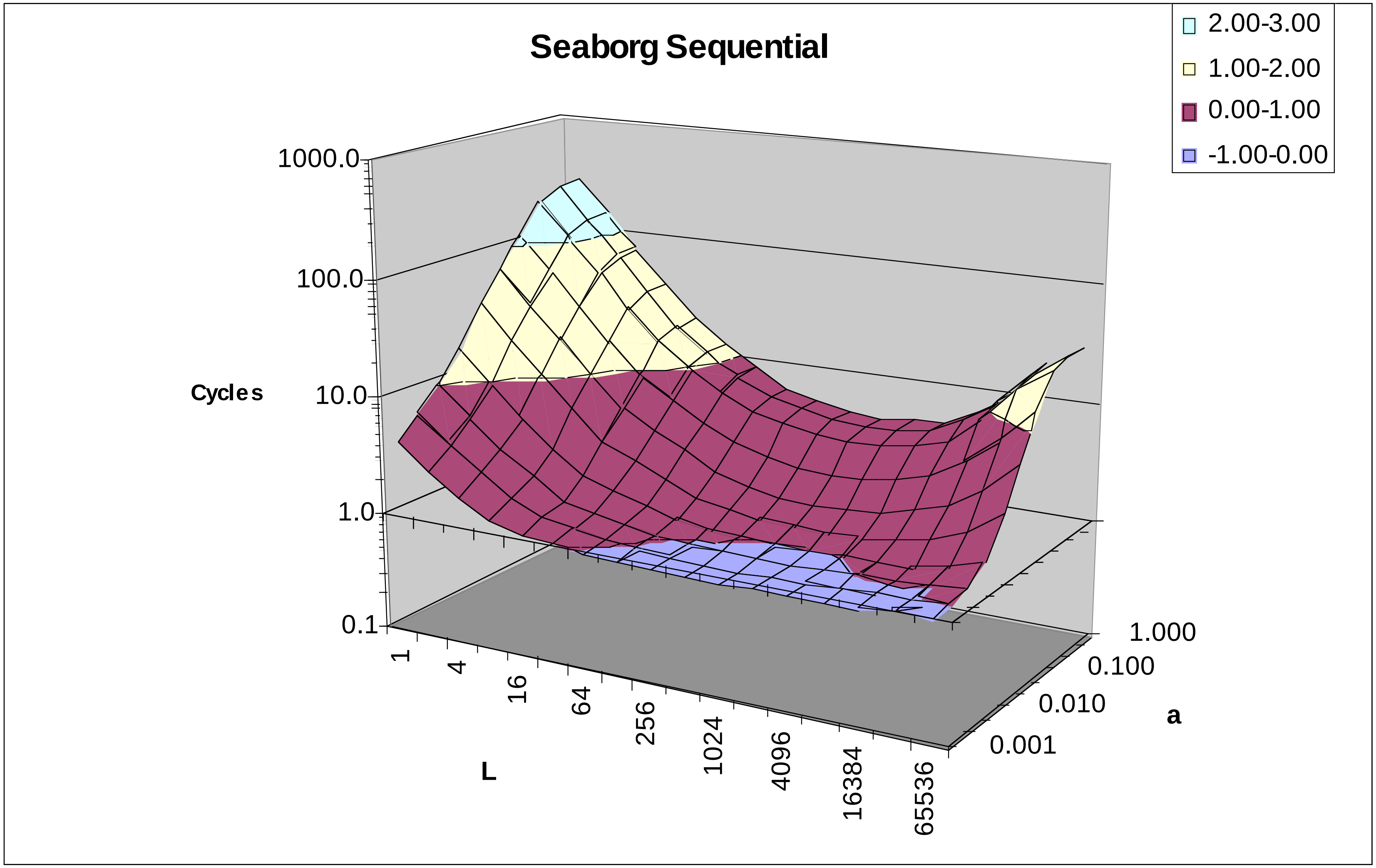






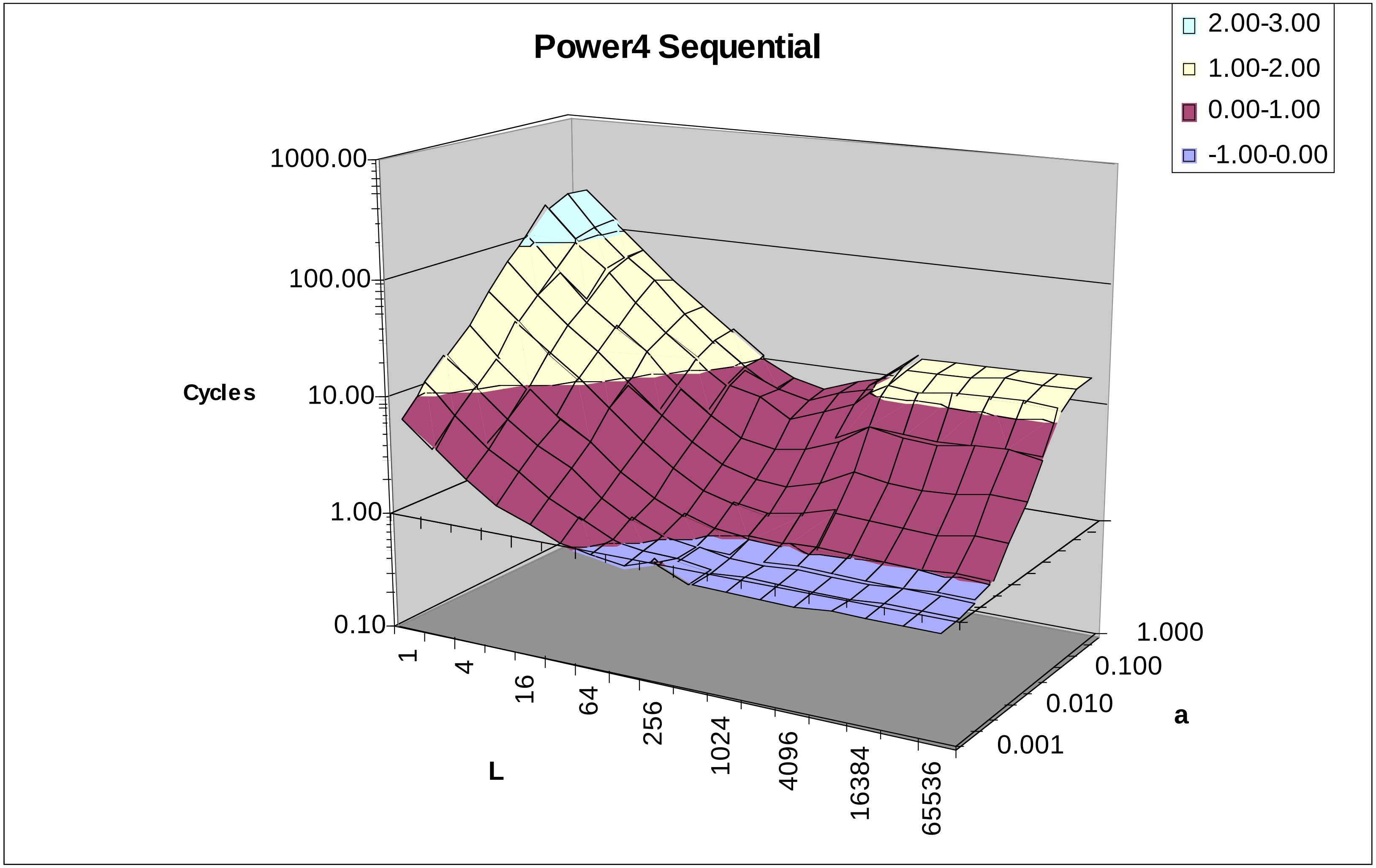
Higher number means higher efficiency without unrolling.

Power3 Sequential



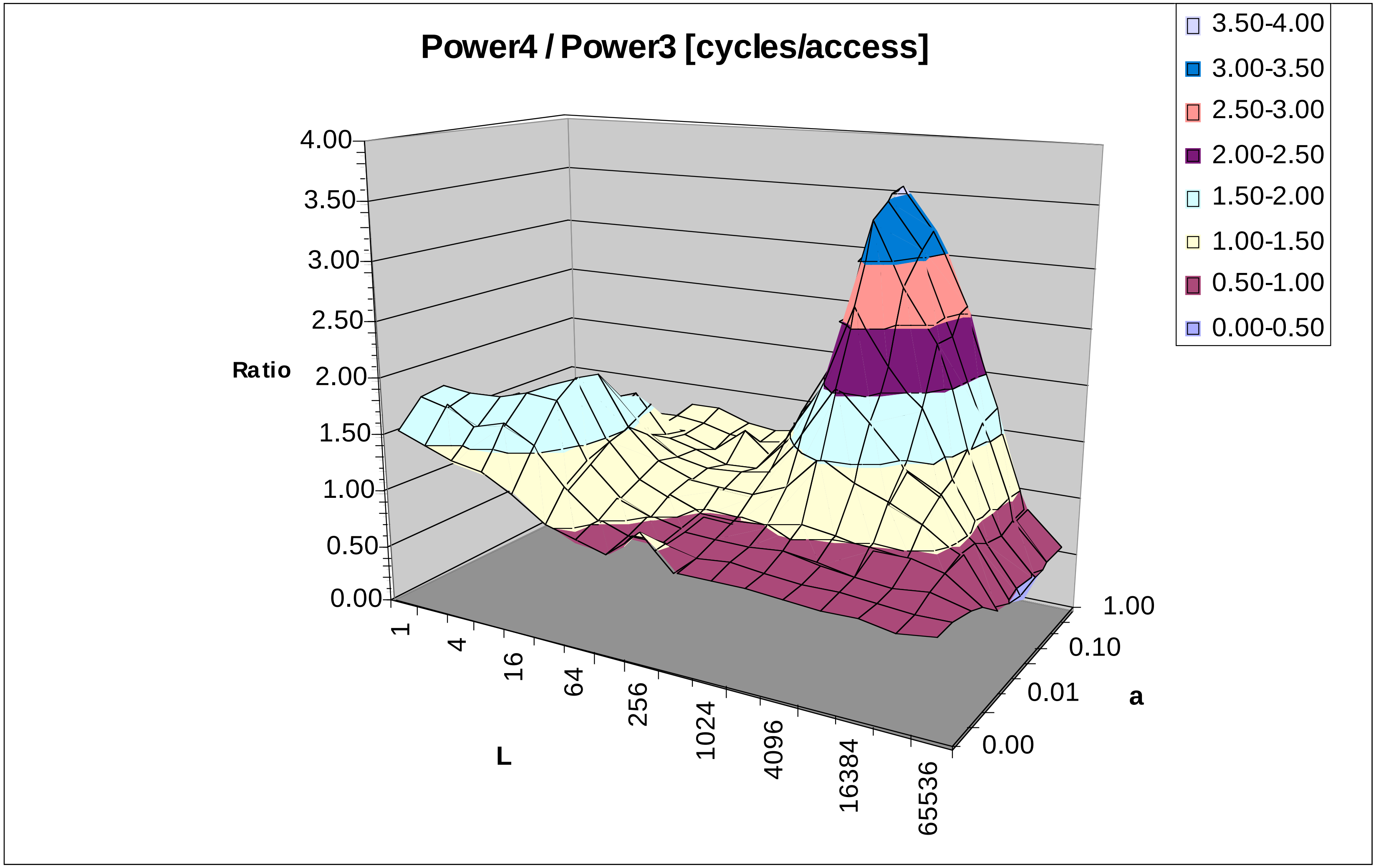
Power3 375 MHz

Power4 Sequential



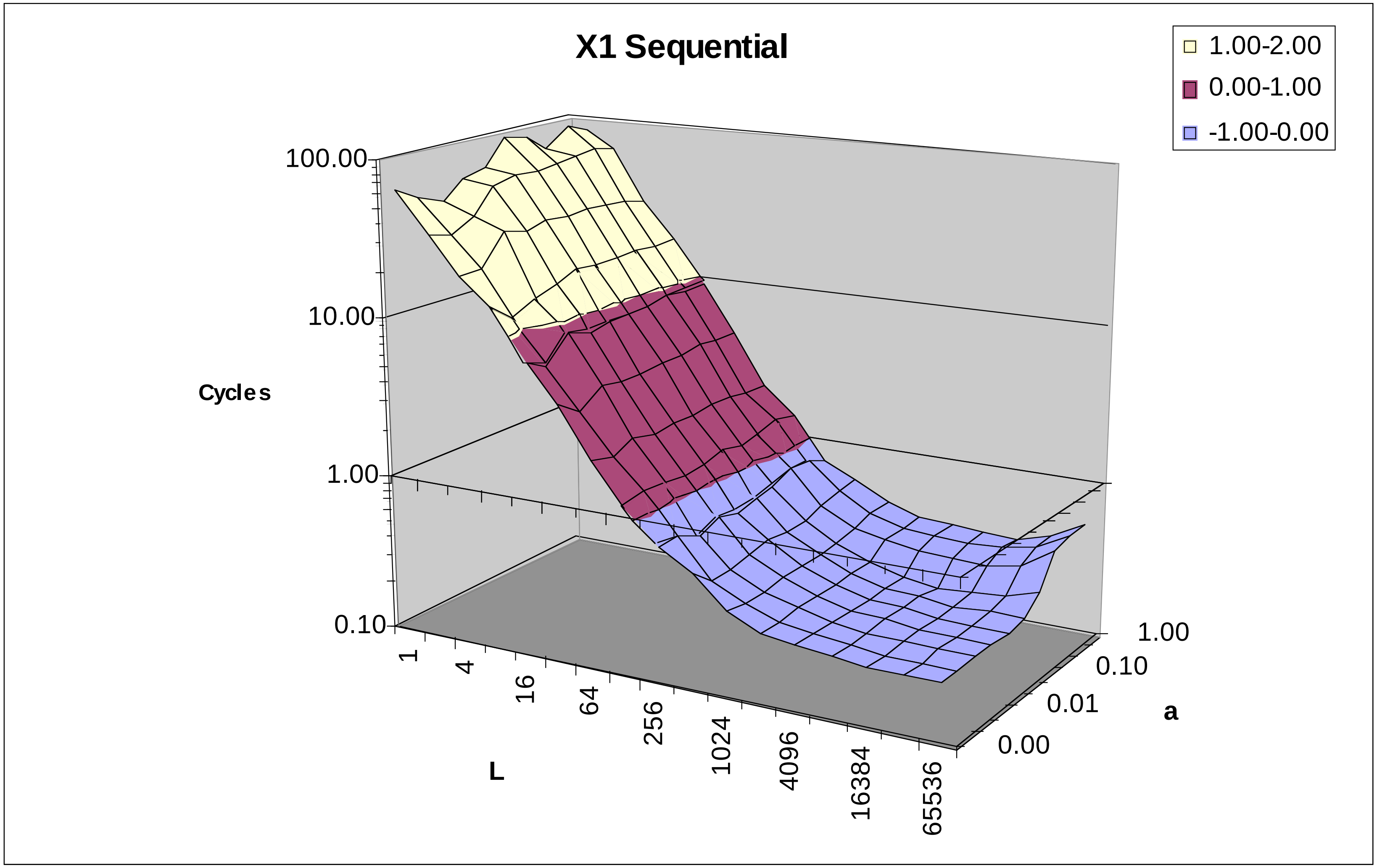
Power4 1300 MHz

Power4 and Power3



Numbers higher than 1 mean higher efficiency for the Power3.

Cray X1



X1 Sequential
1.00-2.00
0.00-1.00
-1.00-0.00
100.00
10.00
1.00
0.10
Cycles
1
4
16
64
256
1024
4096
16384
65536
L
1.00
0.10
0.01
0.00
a

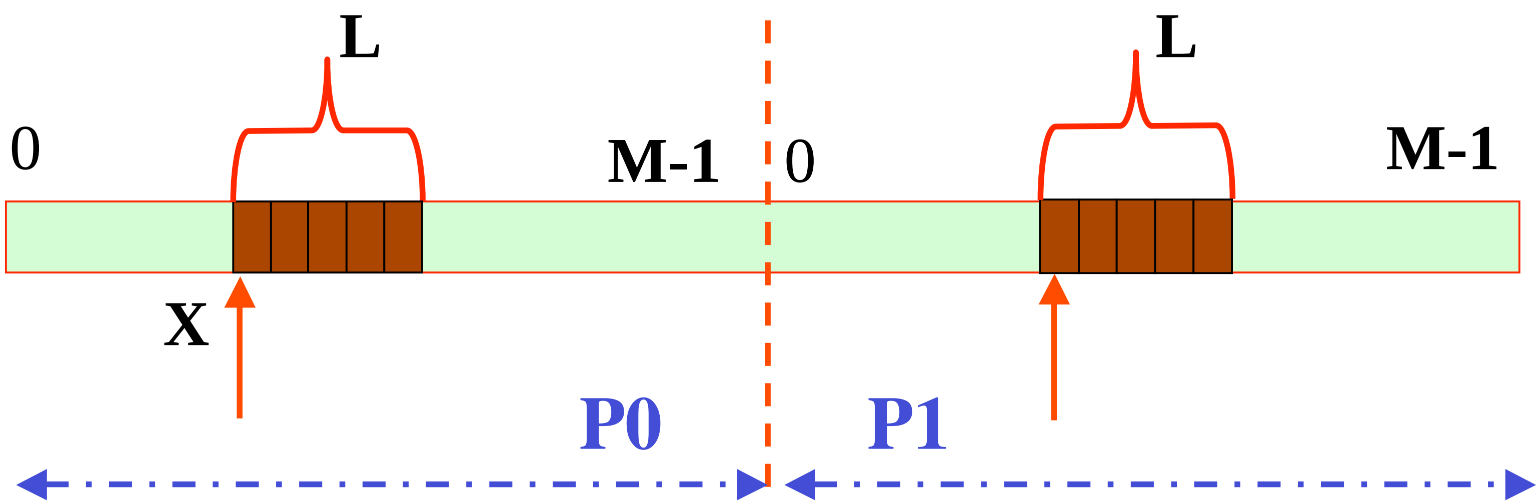
APEX-Map in Parallel

- APEX-Map principals and previous sequential results
- **Parallel APEX-Map and results**
- APEX-Map possible extensions, current and future work

Outline

- **Parallel APEX-Map Design and Implementation**
- APEX-Map helps identifying system performance problems
- APEX-Map helps understanding system performance characteristics
- APEX-Map enables flexible performance comparison
- APEX-Map helps understanding the effects of different programming models
- Summary

APEX-Map Design



- Same parameters as sequential version (α, L, M*P)
- Data evenly distributed among processes
- X distributed across all processes
- Each remote access results in a message with length L (for the message passing paradigm)

APEX-Map Outline

Sequential:

Repeat N Times

Generate Index Array()

CLOCK(start)

For each Index *i* in the Array

Compute()

CLOCK(end)

RunningTime += end - start

End Repeat

Parallel:

Repeat N Times

Generate Index Array()

CLOCK(start)

For each Index *i* in the Array

If (remote data)

GetRemoteData()

End If

Compute()

CLOCK(end)

RunningTime += end - start

End Repeat

Depends on the chosen parallel programming model

SHMEM Implementation

```
Repeat N Times
    Generate Index Array()
    CLOCK(start)
    For each Index i in the Array
        If (remote data)
            SHMEM_DOUBLE_GET(Rid, Offset)
        End If
        Compute()
    CLOCK(end)
    RunningTime += end - start
End Repeat
```

UPC Implementation

Repeat N Times

Generate Index Array()
CLOCK(start)
For each Index *i* in the Array

If (remote data)
UPC_MEMGET(Rid, Offsset)
End if
Compute()
	p = global_data[Rid]+Offset
	for (i = 0; i < L; i++)
	Compute(*(p+i))

CLOCK(end)
RunningTime += end - start

End Repeat

Method 1: Block Access

Method 2: Elementary Access

MPI Implementation

Repeat N Times

Generate Index Array()
CLOCK(start)
For each Index *i* in the Array

If (remote data)
Generate Remote Request()
End If
Serve Incoming Requests()
Compute() if data available

CLOCK(end)
RunningTime += end - start

End Repeat

Computing

CLOCK(start)
Wait For Finish ()
CLOCK(end)
RunningTime += end - start

Waiting

BIPS

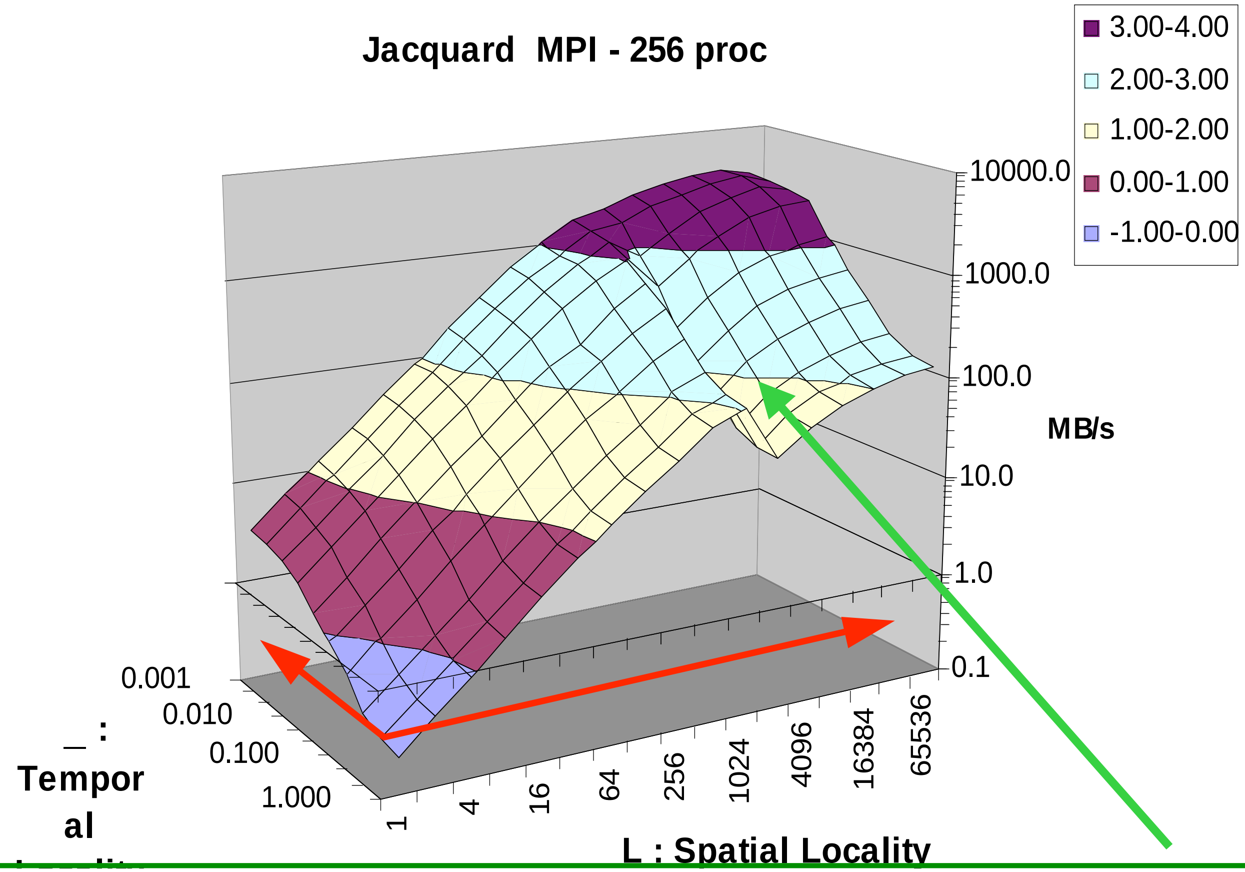
Platforms

	CPU	CPUs/ Node	Network	Memory Bandwidth	Site
Seaborg	Power3 375MHz	16	IBM Colony-II 1GB/s/node	1GB/s/Proc	NERSC
Cheetah	Power4 1300MHz	32	IBM Federation 4GB/s/node	1.37GB/s/Proc	Oak Ridge
BG/L	PowerPC 700MHz	2	3-D Torus, 2.1GB/node Global Tree, 2.1GB/node	5.5GB/s/Proc	Argonne
Phoenix	Cray X1 800MHz	4	2-D Torus 25GB/s/node	25.6GB/s/MSP	Oak Ridge
Jacquard	Opteron 2200MHz	2	Inifinband	6.4GB/s/Proc	NERSC
Thunder	Itanium2 1400MHz	4	Quadrics	6.4GB/s/Proc	LLNL
Altix	Itanium2 1500MHz	2	Fat-tree, NUMALink 6.4GB/link	6.4GB/s/Proc	Oak Ridge

Outline

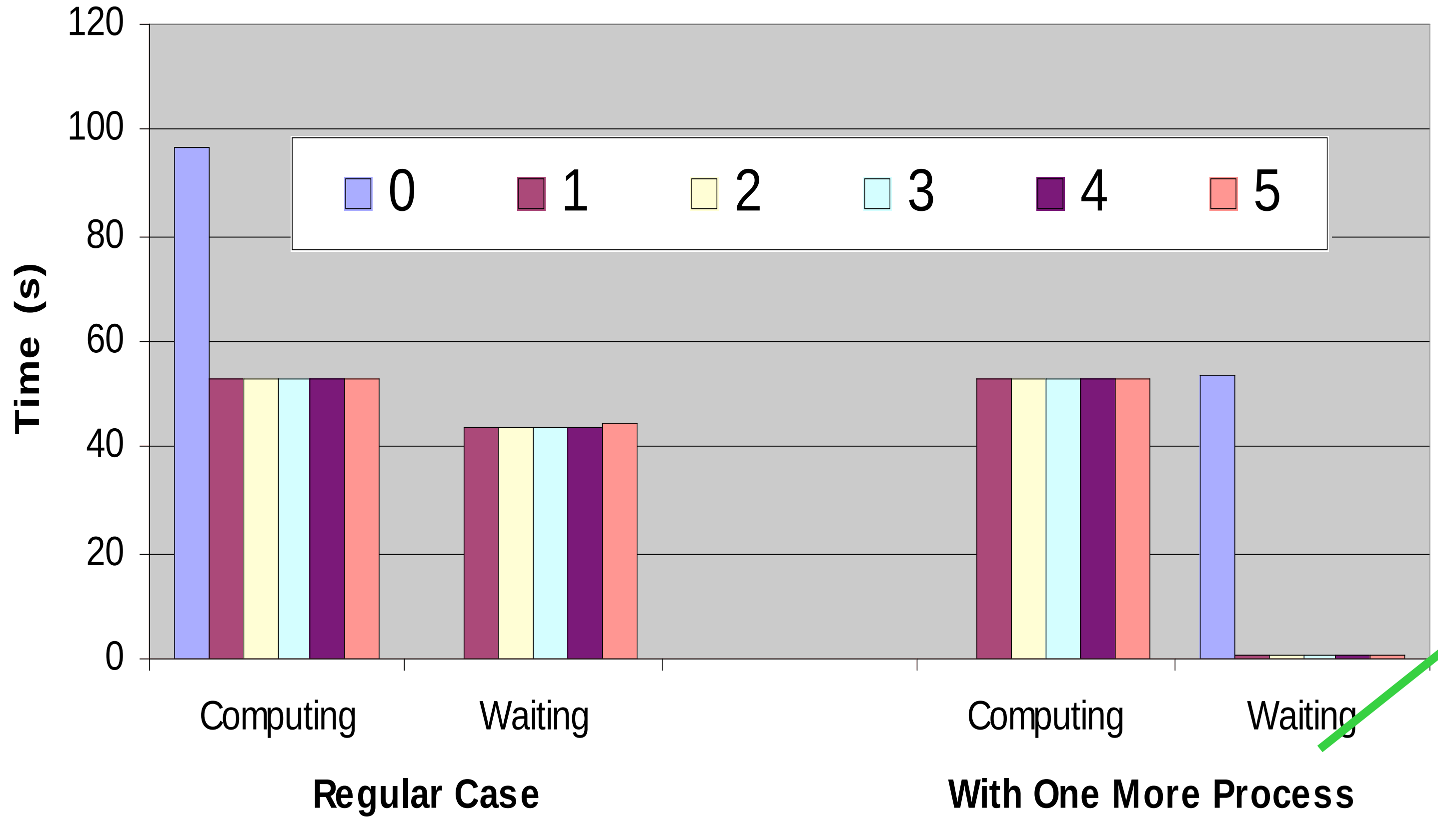
- Parallel APEX-Map Design and Implementation
- **APEX-Map helps identifying system performance problems**
- APEX-Map helps understanding system performance characteristics
- APEX-Map enables flexible performance comparison
- APEX-Map helps understanding the effects of different programming models
- Summary

Jacquard



Performance Drop After L > 1024 Due to Buffer Management

Cray X1 MPI Problem
(α=1, L=1, M = 512MB * 256)



Let process 0 idle

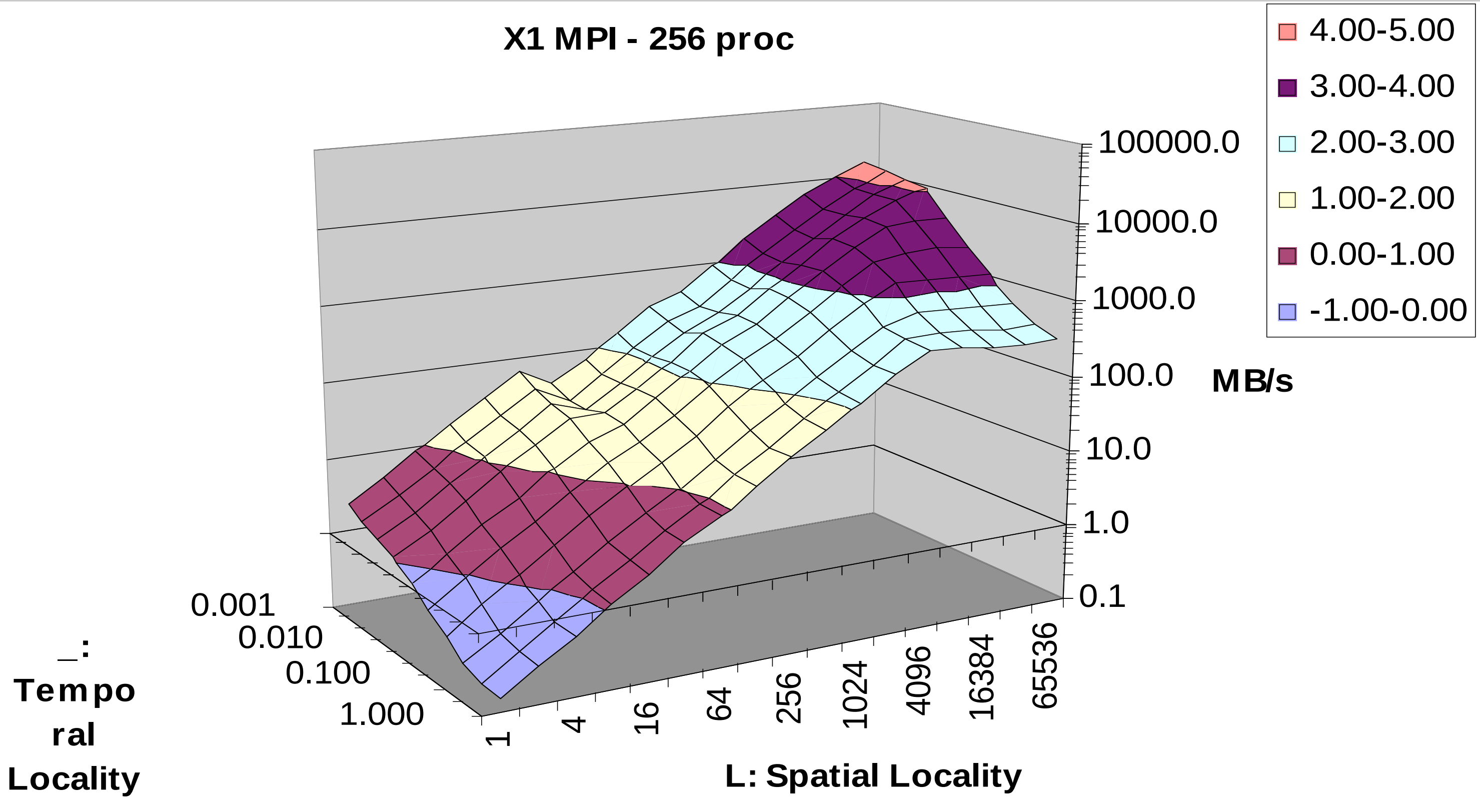
Shift work from i -> i + 1,

Performance Could almost Double for α = 1, L = 1, P = 256

Outline

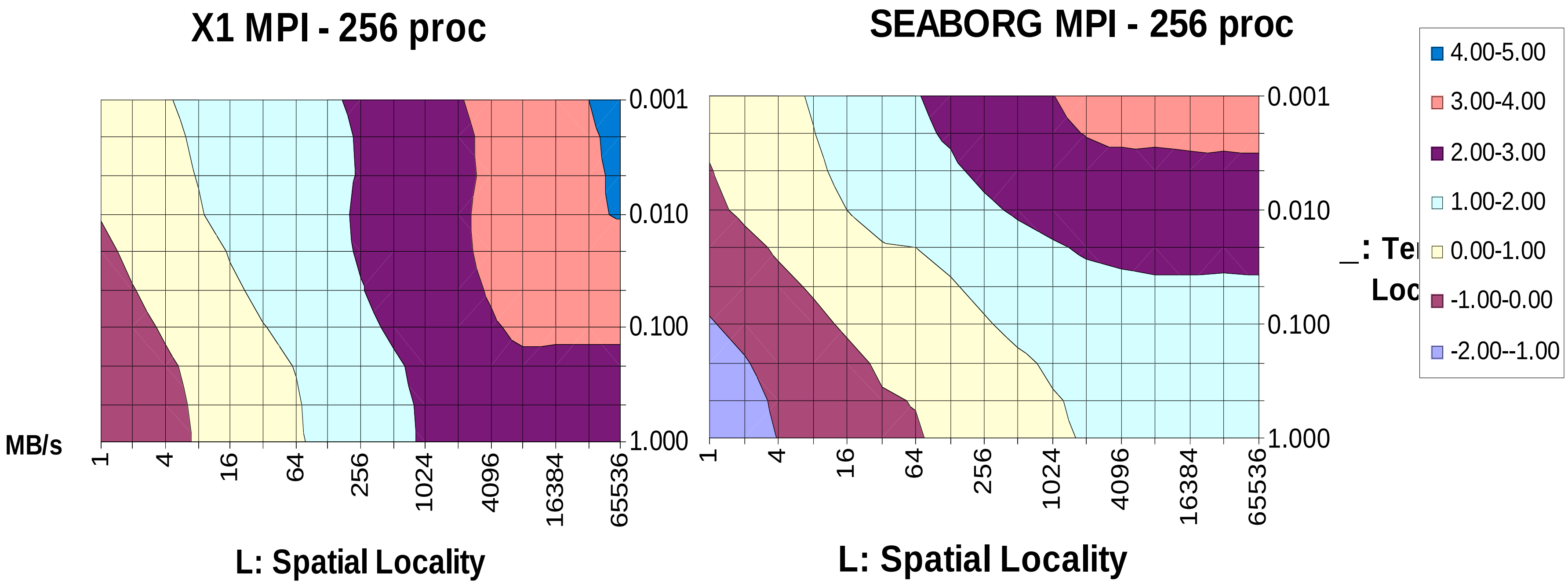
- Parallel APEX-Map Design and Implementation
- APEX-Map helps identifying system performance problems
- **APEX-Map helps understanding system performance characteristics**
- APEX-Map enables flexible performance comparison
- APEX-Map helps understanding the effects of different programming models
- Summary

Performance Surface



- **Both α and L affect the performance**
- **The effects of α and L can to some extent replace each other**

Performance Characteristics (Phoenix vs. Seaborg)

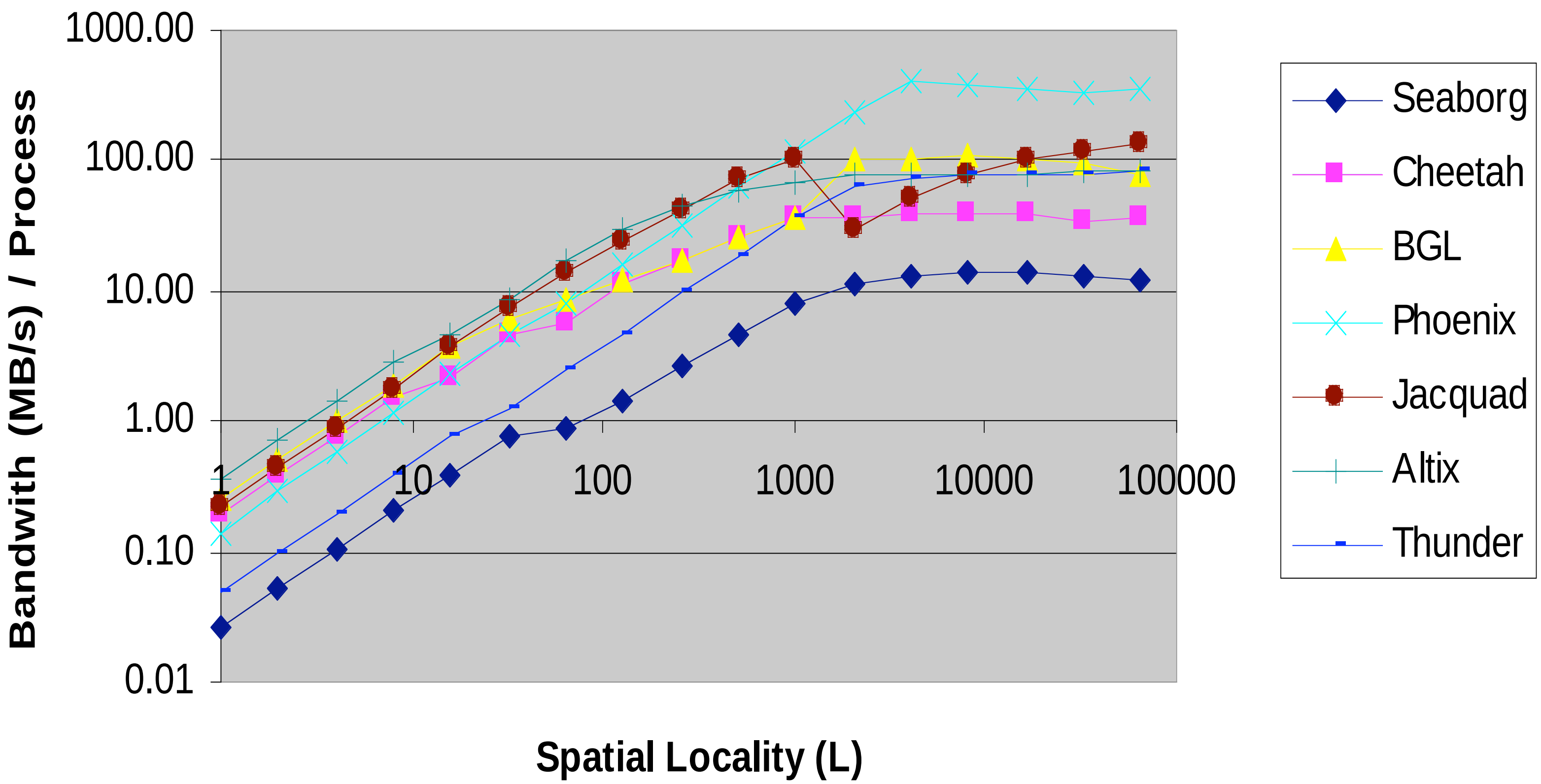


- **Phoenix performance is more sensitive to spatial locality while Seaborg is more sensitive to temporal locality**

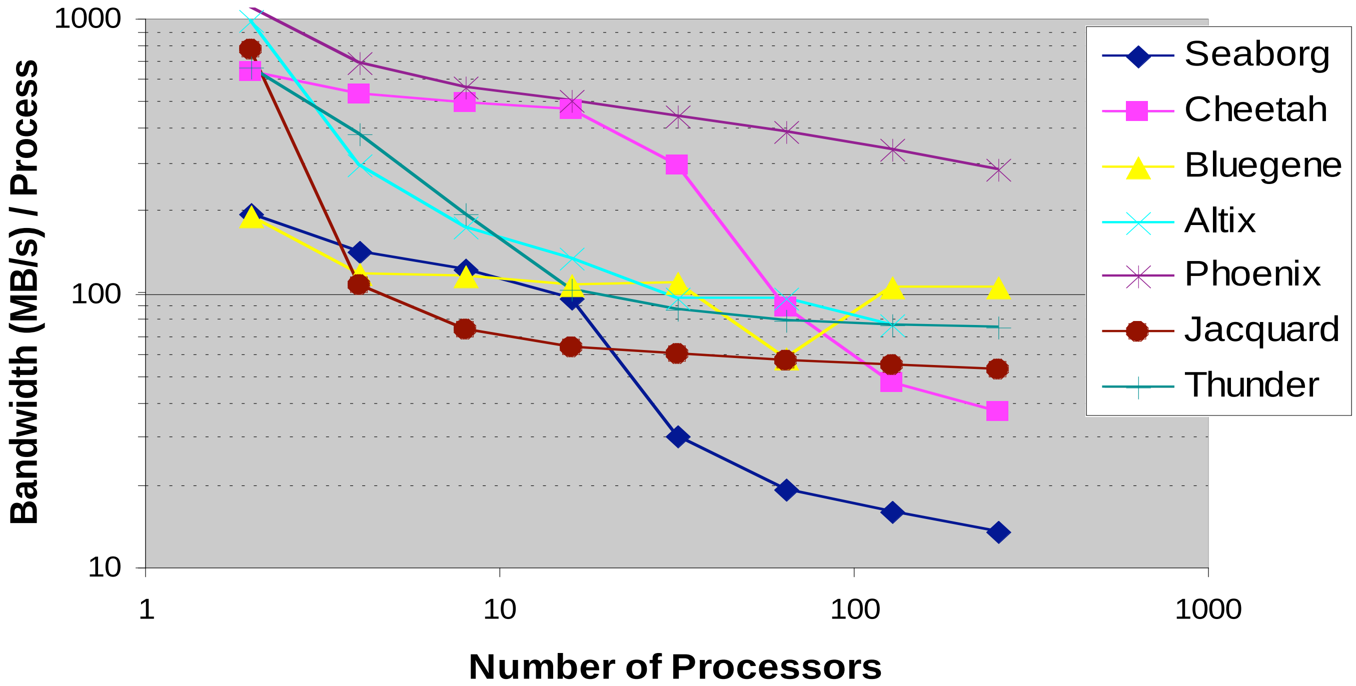
Outline

- Parallel APEX-Map Design and Implementation
- APEX-Map helps identifying system performance problems
- APEX-Map helps understanding system performance characteristics
- **APEX-Map enables flexible performance comparison**
- APEX-Map helps understanding the effects of different programming models
- Summary

Performance Comparison
(α = 1)

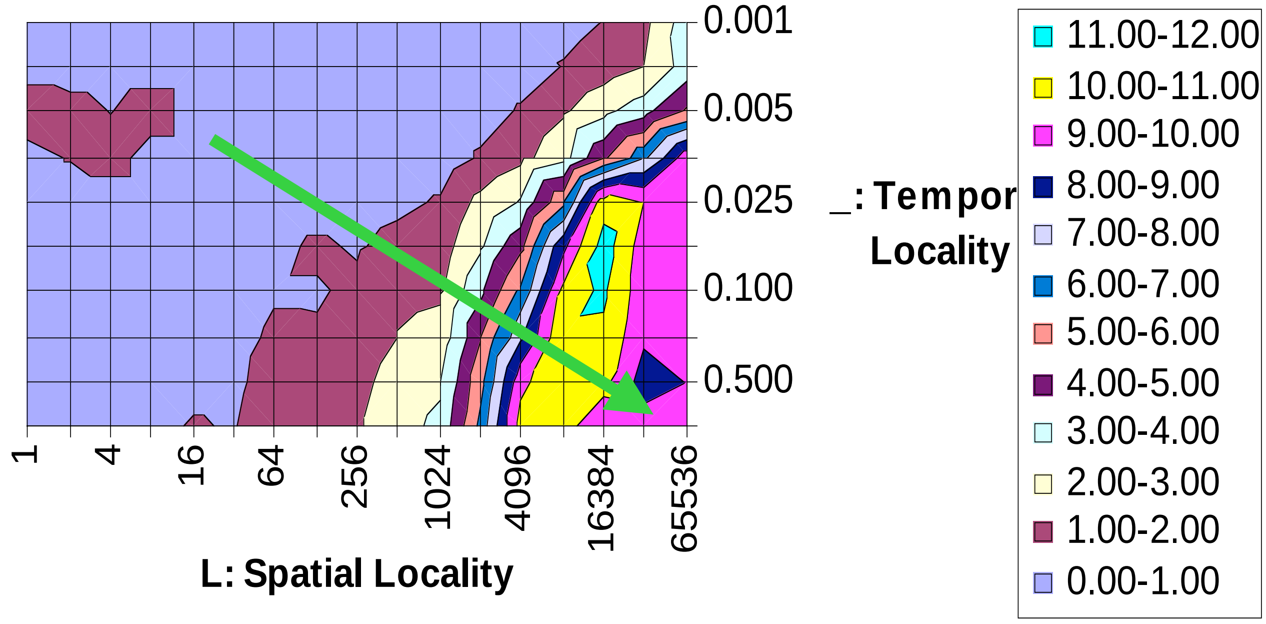


Scalability Analysis (α =1, L=4096)



- **SMP effect**
- **BG/L, Jacquard, Thunder scales better with lower absolute performance than Phoenix**

Direct Comparison (Cray X1 vs. Power4)

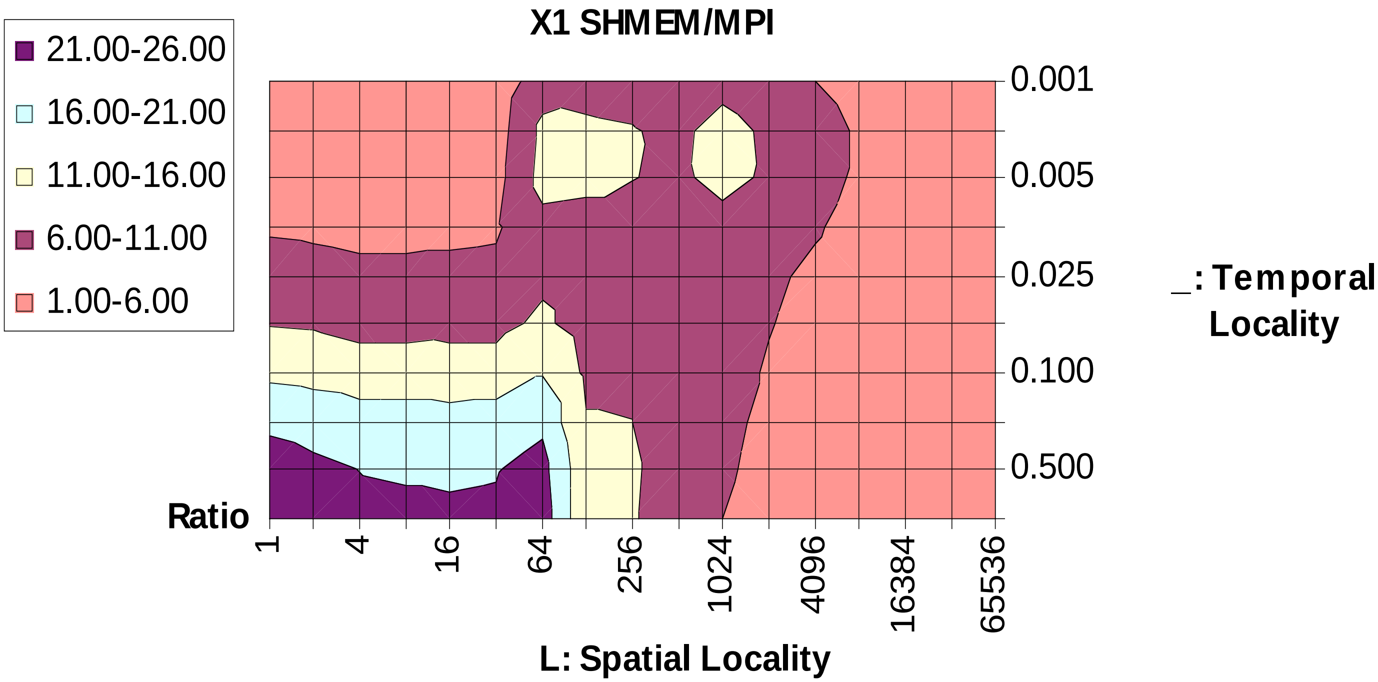


- **Power4 performs better for smaller L and smaller α**
- **Cray X1 is much more efficient at bottom-right corner**

Outline

- Parallel APEX-Map Design and Implementation
- APEX-Map helps identifying system performance problems
- APEX-Map helps understanding system performance characteristics
- APEX-Map enables flexible performance comparison
- **APEX-Map helps understanding the effects of different programming models**
- Summary

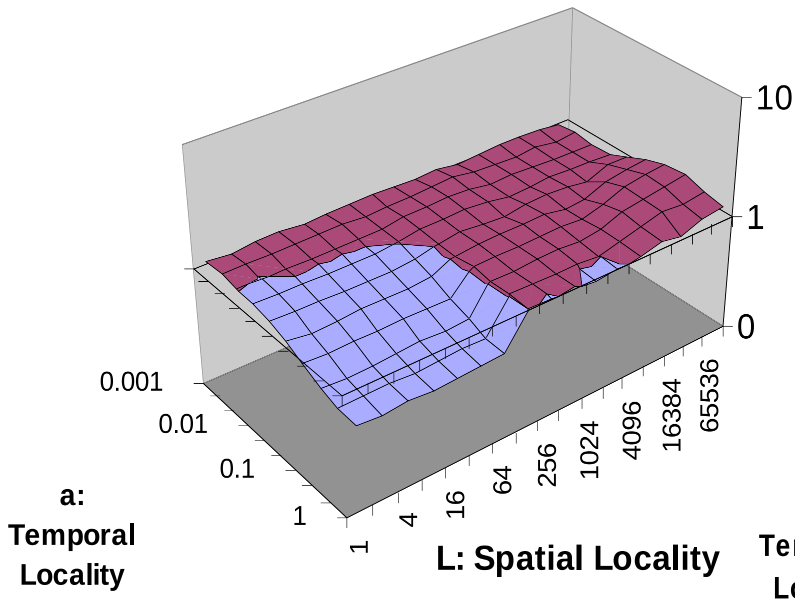
Cray X1 SHMEM / MPI



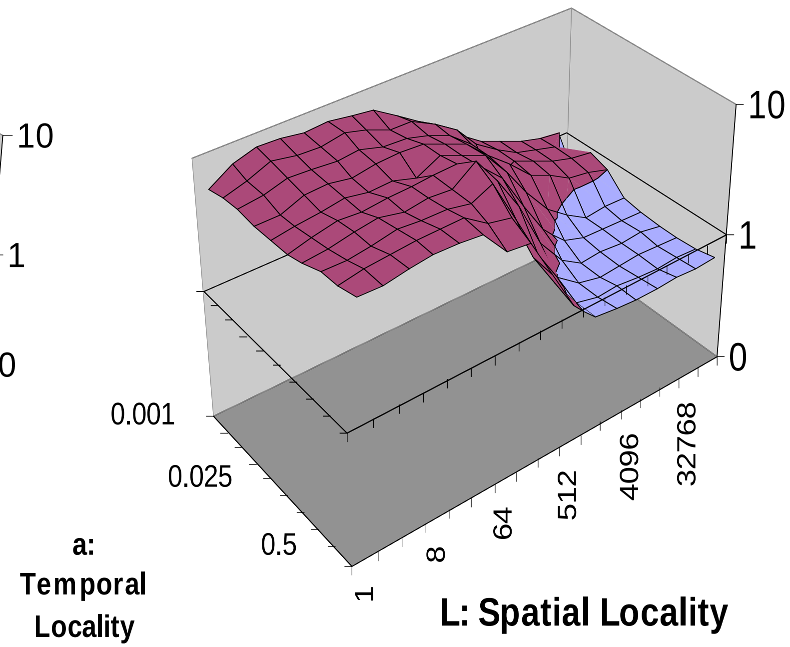
- **The advantage of SHMEM is much stronger for low temporal locality and low spatial locality**

Cray X1 : UPC / SHMEM

Method 1: Block Access



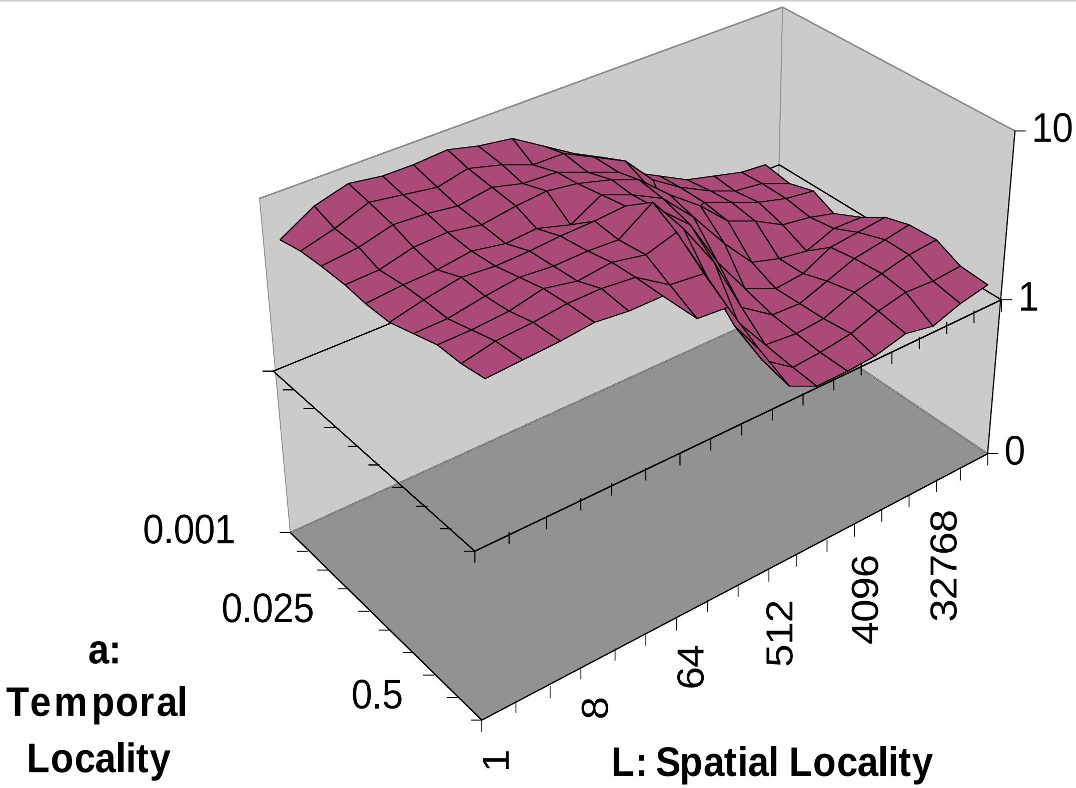
Method 2: Elementary Access



- Block transfer is efficient for large data blocks
- Regular load is efficient for short data blocks

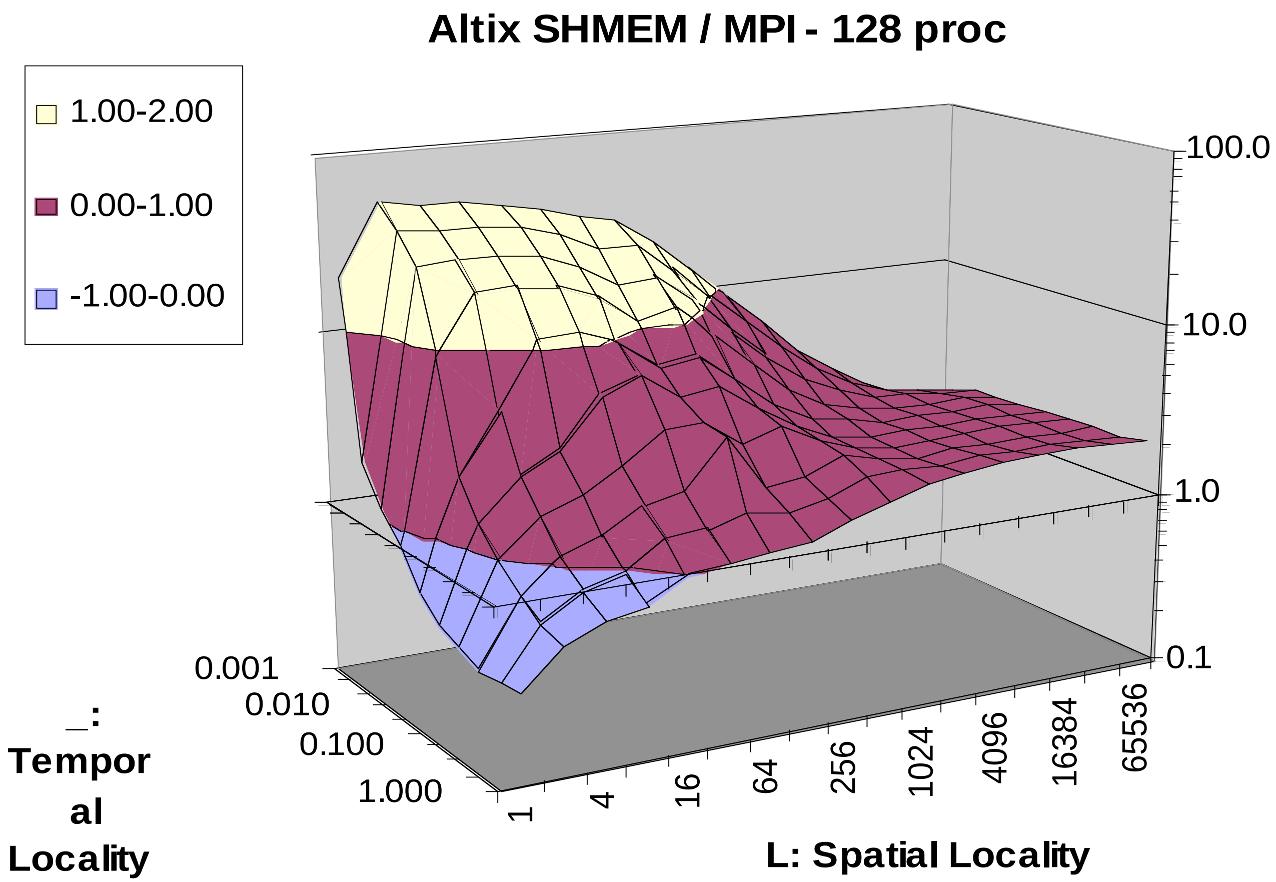
Office of Science
U.S. DEPARTMENT OF ENERGY

Cray X1: UPC / SHMEM



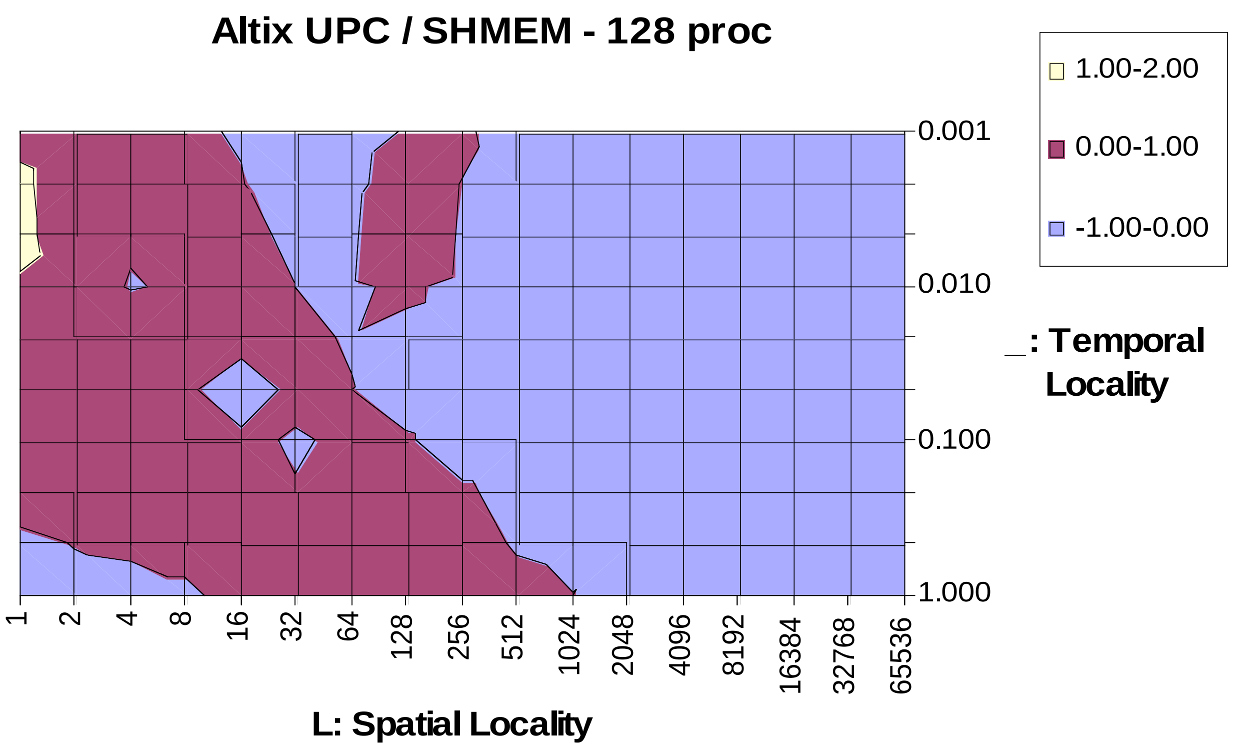
- UPC is better than SHMEM if we choose best implementation for UPC for all cases
- Performance: UPC > SHMEM > MPI

Altix: SHMEM / MPI



- **SHMEM not always better than MPI**

Altix: UPC / SHMEM



- **UPC also not always better than SHMEM**
- **UPC is basically efficient for short messages**

Summary

- APEX-Map helps to identify performance problems on Cray X1 and Jacquard
- APEX-Map enables flexible performance comparison and understanding of performance characteristics
- APEX-Map can be used to analyze the effects of different programming models

APEX-Map Extensions

- APEX-Map principals and previous sequential results
- Parallel APEX-Map and results
- **APEX-Map possible extensions, current and future work**

APEX-Map Extensions

- **Extending the data stream model to differentiate the effects of stride 1 access and of vectorization**
  - Additional parameter to describe address streams
- **Insert a parameterized compute kernel to study performance effects of register pressure and computational intensity**
  - Two new computational parameters

APEX-Map Extensions

Possible ideas for extending APEX-Map

- Research connection of locality definitions to other definitions proposed (HPCS)
  - SC05 Paper with SDSC
- Alternative implementations, paradigms, and optimizations for parallel version
- Extend to new architectures and new paradigms (HPCS vendors)
  - Run APEX-Map on simulators (Testing on Sigma now)
- Investigate a network topology sensitive mode of APEX-Map

APEX-Map Extensions

- Work with other benchmarking teams, application experts (SCIDAC and HPCS in particular) to determine where and how their applications fit on the performance landscape
- Research what the main factors for performance differences between APEX-Map and applications are.
- Performance Prediction for codes beyond the point of measurability
- Analyze how compilers affect measured performances (compiler shortcoming)
- Research implications for performance optimization

BIPS

Conclusions

- It is possible to explore the full performance space with a single code.
- APEX-Map allows to map performance for the whole range of temporal and spatial localities.
- Very regular codes might be hard to approximate with the random version of APEX-Map.
- http://ftg.lbl.gov

APEX-Map Publications - FY06

Publications (E. Strohmaier, H. Shan):

- **CUG2005:** ***MPI, SHMEM, and UPC Performance on the Cray X1 — A Case Study using APEX-Map***
- **EuroPar 2005:** ***Apex-Map: A Synthetic Scalable Benchmark Probe to Explore Data Access Performance on Highly Parallel Systems***
- **SC2005:** ***Apex-Map: A Global Data Access Benchmark to Analyze HPC Systems and Parallel Programming Paradigms***
- **SC2005:** ***Measurement of Spatial and Temporal Locality in Memory Access Patterns*****, J. Weinberg, A. Snavely, M.O. McCracken, E. Strohmaier**

Additional Talks:

- **LACSI 2004**
- **SC2004, TOP500 BoF**
- **HPCS Productivity Meeting 2005**
- **SDSC Seminar**
- **HPCS PI Meeting (Pedro Diniz, ISI)**